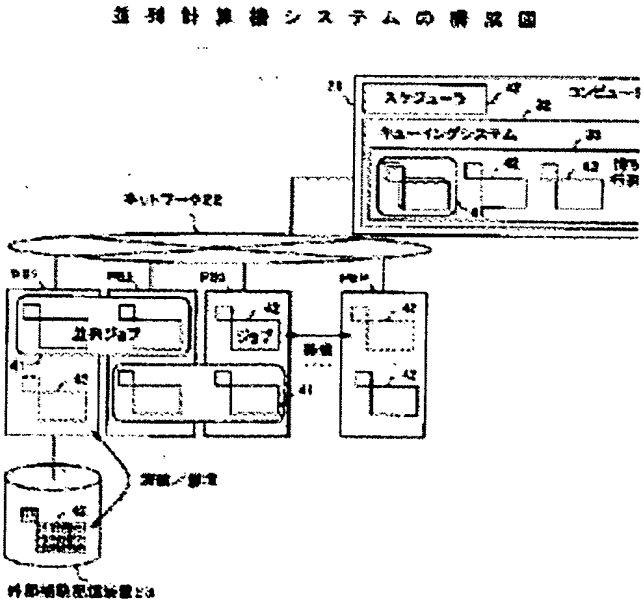
SCHEDULING DEVICE FOR PERFORMING JOB SCHEDULING OF PARALLEL-COMPUTER SYSTEM

Publication number: JP2002007364
Publication date: 2002-01-11
Inventor: YAMASHITA KOICHIRO
Applicant: FUJITSU LTD
Classification:
- International: ***G06F15/177; G06F9/50;*** **G06F15/16; G06F9/46;** (IPC1-7): G06F15/177; G06F9/46
- European: G06F9/46A4M
Application number: JP20000187317 20000622
Priority number(s): JP20000187317 20000622

Also published as:
US7024671 (B2)
US2002002578 (A

Abstract of **JP2002007364**
PROBLEM TO BE SOLVED: To improve the utilization efficiency of computer resources by reducing a scheduling error accompanying job execution in a parallel-computer system. SOLUTION: Jobs 41 and 42 inputted from a queuing system 32 to a processor element(PE) are moved to the other PE or frozen and written out to an external auxiliary storage device 23 by dynamic scheduling by a scheduler 31. The scheduler 31 estimates the remaining execution time of the respective jobs and determines the job of a moving/freezing target in comparison with moving/ freezing costs.



(19)日本国特許庁（ＪＰ）　(12) **公開特許公報**（Ａ）　(11)特許出願公開番号
特開2002－7364
（P2002－7364A）
(43)公開日　平成14年１月11日(2002.1.11)

(51)Int.Cl.⁷	識別記号	ＦＩ	テーマコード（参考）
Ｇ０６Ｆ　15/177	６７４	Ｇ０６Ｆ　15/177	６７４Ａ　５Ｂ０４５
			６７４Ｂ　５Ｂ０９８
	６７２		６７２Ｂ
9/46	３６０	9/46	３６０Ｂ
			３６０Ｃ

審査請求　未請求　請求項の数５　ＯＬ　（全 17 頁）

(21)出願番号　特願2000－187317(P2000－187317)

(22)出願日　平成12年６月22日(2000.6.22)

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(72)発明者　山下　浩一郎
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

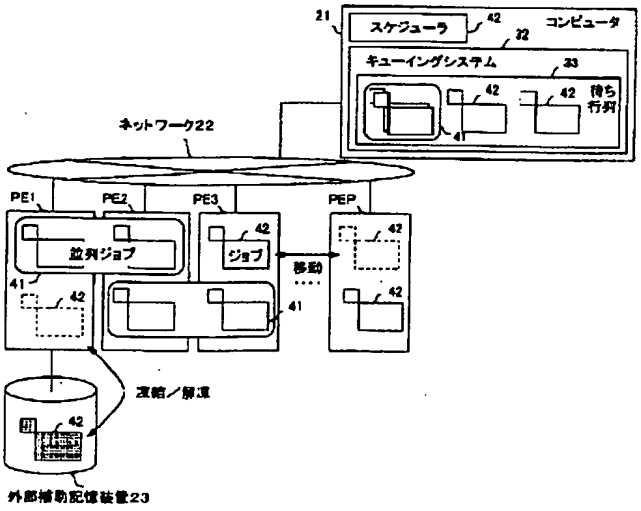
(74)代理人　100074099
弁理士　大菅　義之　（外１名）

Ｆターム(参考)　5B045 JJ08
5B098 AA10 GA03 GA08 GC10 GD02

(54)【発明の名称】　並列計算機システムのジョブスケジューリングを行うスケジューリング装置

(57)【要約】
【課題】　並列計算機システムにおいて、ジョブ実行に伴うスケジューリング誤差を削減し、計算機資源の利用効率を向上させることが課題である。
【解決手段】　キューイングシステム３２からプロセッサエレメント（ＰＥ）に投入されたジョブ４１、４２は、スケジューラ３１による動的スケジューリングにより、他のＰＥに移動したり、凍結されて外部補助記憶装置２３に書き出されたりする。スケジューラ３１は、各ジョブの実行残り時間を推定し、移動／凍結コストと比較して、移動／凍結対象のジョブを決定する。

並列計算機システムの構成図



【特許請求の範囲】
【請求項１】　複数のプロセッサエレメントを有する並列計算機システムのジョブスケジューリングを行うスケジューリング装置であって、
プロセッサエレメントで実行中のジョブを他のプロセッサエレメントに移動するべきか否かを判定する判定手段と、
前記ジョブが前記他のプロセッサエレメントに移動するべきであると判定されたとき、該ジョブのマイグレーション処理が行われるように、前記複数のプロセッサエレメントに対する実行中のジョブの割当を行う割当手段とを備えることを特徴とするスケジューリング装置。
【請求項２】　前記複数のプロセッサエレメントの負荷状況を監視する監視手段をさらに備え、前記複数のプロセッサエレメント間で負荷分布がアンバランスになったとき、前記割当手段は、前記実行中のジョブの割当を行うことを特徴とする請求項１記載のスケジューリング装置。
【請求項３】　前記判定手段は、前記実行中のジョブの情報を含むジョブ情報テーブルを生成し、該ジョブ情報テーブルのジョブのうち、移動するジョブを決定して、前記複数のプロセッサエレメント上に再配置されたジョブの情報を含む再配置リストを生成し、前記割当手段は、該再配置リストに基づいて、該実行中のジョブの割当を行うことを特徴とする請求項１記載のスケジューリング装置。
【請求項４】　前記判定手段は、前記ジョブのマイグレーション処理に要するコストを計算し、計算されたコストに基づいて、前記ジョブを前記他のプロセッサエレメントに移動するべきか否かを判定することを特徴とする請求項１記載のスケジューリング装置。
【請求項５】　複数のプロセッサエレメントを有する並列計算機システムのジョブスケジューリングを行うコンピュータのためのプログラムを記録した記録媒体であって、
前記プログラムは、
プロセッサエレメントで実行中のジョブを他のプロセッサエレメントに移動するべきか否かを判定し、
前記ジョブが前記他のプロセッサエレメントに移動するべきであると判定されたとき、該ジョブのマイグレーション処理が行われるように、前記複数のプロセッサエレメントに対する実行中のジョブの割当を行う処理を前記コンピュータに実行させることを特徴とするコンピュータ読み取り可能な記録媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、並列計算機システムにおいて、動的にジョブスケジューリングを行うスケジューリング装置に関する。
【０００２】
【従来の技術】従来の並列計算機システムにおいては、ＣＰＵ（中央処理装置）やメモリ等の計算機資源を効率良く利用するために、ジョブのスケジューリングが行われている。ここでは、計算機システムにおいて実行されるプログラムの最小単位をプロセスと定義し、プロセスの複合体として並列実行される単位をジョブと定義することにする。
【０００３】ジョブの実行待ち行列の管理等のスケジューリング方法としては、主に以下のようなものが用いられている。
（１）ＦＩＦＯ（First In First Out）によるジョブの連続投入／実行操作
（２）優先度による待ち行列内のジョブのソート（スケジューリング）操作
このうち、（２）の方法では、各ジョブに優先度フラグ等を付加しておき、後から待ち行列に投入されたジョブであっても、優先度が高ければ、待ち行列の先頭に割り込ませるような制御を行う。さらに、これらの方法に加えて、ジョブの投入に際して、様々なスケジューリング方法が提案されている。
【０００４】
【発明が解決しようとする課題】しかしながら、上述した従来のスケジューリング方法には、ジョブ投入後の不確定要素に対処できないという問題がある。
【０００５】ジョブが実行される場合、実際の実行時間を実行前に正確に把握することは困難である。これは、ＯＳ（オペレーティングシステム）による制御を前提とした場合に、ジョブの実行が、プロセスのスイッチング、あるプロセスと他のプロセスの資源の奪い合い、外部との入出力等の不確定要素を含んでいるためである。
【０００６】この場合、実行時間を正確に管理していないスケジューリング方法によりスケジューリングを行うと、ジョブの実行が進むにつれて、スケジューリングの誤差が大きくなる。その結果、特定のＰＥ（プロセッサエレメント）に負荷が偏る等の事態が発生し、並列計算機システムの運用が非効率的になる。また、ＯＳが複数のプロセスを管理している以上、このような不確定要素によるスケジューリング誤差の発生は不可避である。
【０００７】本発明の課題は、並列計算機システムにおいて、ジョブ実行に伴うスケジューリング誤差を削減し、計算機資源の利用効率を向上させるスケジューリング装置を提供することである。
【０００８】
【課題を解決するための手段】図１は、本発明のスケジューリング装置の原理図である。本発明の第１の局面において、スケジューリング装置は、判定手段１１と割当手段１２を備え、複数のＰＥを有する並列計算機システムのジョブスケジューリングを行う。
【０００９】判定手段１１は、あるＰＥで実行中のあるジョブが他のＰＥにマイグレート（移動）するべきか否

かを判定する。割当手段１２は、そのジョブが他のＰＥにマイグレートするべきであると判定されたとき、そのジョブのマイグレーション処理が行われるように、複数のＰＥに対する実行中のジョブの割当を行う。

【００１０】ジョブのマイグレーション処理とは、あるＰＥで実行中のジョブを動作させたままで他のＰＥに移動させる処理を表す。判定手段１１は、あるジョブが他のＰＥにマイグレートするべきか否かを判定し、判定結果を割当手段１２に渡す。割当手段１２は、そのジョブが他のＰＥにマイグレートするべきであるという判定結果を受け取ると、そのジョブを他のＰＥに割り当てるとともに、他の実行中のジョブの再割当を行う。

【００１１】このようなスケジューリング装置によれば、ジョブのマイグレーション処理を考慮したジョブスケジューリングが行われるので、ジョブ実行に伴うスケジューリング誤差が発生したときでも、ＰＥ上のジョブの配置を適切に変更することができる。したがって、スケジューリング誤差が削減され、計算機資源の利用効率が向上する。

【００１２】また、本発明の第２の局面において、スケジューリング装置は、静的スケジューリング手段１３と動的スケジューリング手段１４を備え、複数のＰＥを有する並列計算機システムのジョブスケジューリングを行う。

【００１３】静的スケジューリング手段１３は、静的スケジューリングを行い、動的スケジューリング手段１４は、静的スケジューリングと組み合わされた動的スケジューリングを行う。

【００１４】静的スケジューリング手段１３は、例えば、静止状態にあるジョブのスケジューリング、または並列計算機システムの構成が固定されている場合のスケジューリングを行う。また、動的スケジューリング手段１４は、例えば、動作状態にあるジョブのスケジューリング、または並列計算機システムの構成が変更可能な場合のスケジューリングを行う。

【００１５】このようなスケジューリング装置によれば、静的スケジューリングと動的スケジューリングを組み合わせた総合的なジョブスケジューリングが行われる。したがって、ジョブ実行に伴うスケジューリング誤差が発生したときでも、ＰＥ上のジョブの配置を動的に変更することで誤差が削減され、計算機資源の利用効率が向上する。

【００１６】例えば、図１の判定手段１１、割当手段１２、静的スケジューリング手段１３、および動的スケジューリング手段１４は、後述する図２のスケジューラ３１に対応する。

【００１７】

【発明の実施の形態】以下、図面を参照しながら、本発明の実施の形態を詳細に説明する。本実施形態においては、ジョブスケジューリングに対して、「静的」および「動的」という概念を取り入れる。そして、ジョブの静止状態あるいはシステムの静止（固定）状態に関連するスケジューリングを「静的スケジューリング」と呼び、ジョブの動作状態あるいはシステムの変動状態に関連するスケジューリングを「動的スケジューリング」と呼ぶことにする。

【００１８】ここで、ジョブの静止状態とは、ジョブが実行待ち行列内にある状態や、ディスク等の外部補助記憶装置に格納されている状態（凍結状態）に対応し、ジョブの動作状態とは、並列計算機システム上でジョブが実行されている状態に対応する。また、システムの静止状態とは、システム構成が、運用中に変更されることなく、安定して稼働している状態に対応し、システムの変動状態とは、システム構成が、保守等により運用中に逐次変更される状態に対応する。

【００１９】静的スケジューリングにより待ち行列内のジョブを組み替えるだけでなく、動的スケジューリングにより、動作中のジョブを凍結したり、移動したりする操作を考慮したスケジューリングを行うことで、計算機資源をより有効に活用することができる。そこで、本実施形態では、計算機資源を有効に活用するために、以下のような方法を採用する。

（１）動的スケジューリングにより、実行時間等の不確定要素によるスケジューリング誤差を吸収する。

（２）動的スケジューリングを効果的に行うために、待ち行列や外部補助記憶装置に対する静的スケジューリングを行う。

（３）動的スケジューリングを用いて並列計算機システム上のジョブの配置を変えることにより、一部のＰＥ資源を解放し、システムの部分保守を行う。

（４）動的および静的スケジューリングを組み合わせた総合的なスケジューリングを行う。

（５）過去のジョブ動作状況のデータを蓄積して、現在のジョブの実行時間を予測する。

【００２０】また、スケジューリングによるジョブの実行配置の最適化問題は、一般に、ＮＰ困難（non-polynomial hard ）な問題であり、最適解を求めるための有効な公式が存在しない。そこで、本実施形態では、経験的（ヒューリスティック）な解法による近似最適解に基づいて、スケジューリングを行う。

【００２１】このようなジョブスケジューリングを実現するために、以下のようなハードウェアおよびソフトウェアを備えた並列計算機システムを想定する。・ハードウェア

（１）ジョブを並列に実行可能なＰＥを複数備えたシステム

（２）各ＰＥを連結するネットワークシステム

（３）ディスク装置等の外部補助記憶装置とその入出力機構・ソフトウェア

（１）複数のプロセスを制御（実行）可能なＯＳ

（２）動作中のジョブを凍結させ、ファイルイメージとして外部補助記憶装置に書き出すようなジョブ凍結機能
（３）動作中のジョブを別のＰＥへ移動させるジョブマイグレーション機能
（４）ジョブの投入／実行を待ち行列により管理するキューイングシステム
（５）ジョブの並列度を知るインタフェース機能
（６）ジョブのメモリサイズを知るインタフェース機能
（７）ジョブの実行優先度を指定するインタフェース機能
ここで、ジョブの凍結とは、ジョブの実行を中断して、その時点の動作状態を表すデータ（ジョブに関する一部または全部の情報）をファイルにパッキングし、ＰＥの外部に保存することを意味する。また、ジョブの並列度は、ジョブの実行に必要なＰＥの数を表し、ジョブのメモリサイズは、ジョブが使用するメモリ量を表す。

【００２２】図２は、このような並列計算機システムの構成図である。図２のシステムは、コンピュータ２１とＰ個のＰＥ（ＰＥ１，ＰＥ２，ＰＥ３，．．．，ＰＥＰ）を備え、これらの装置はネットワーク２２により互いに接続されている。各ＰＥには、必要に応じて、外部補助記憶装置２３を接続することができるが、ここでは、ＰＥ１に外部補助記憶装置２３が接続されている。外部補助記憶装置２３としては、例えば、磁気ディスク装置、光ディスク装置、光磁気ディスク（magneto-optical disk）装置、テープ装置等が用いられる。

【００２３】コンピュータ２１は、スケジューラ３１とキューイングシステム３２を備える。キューイングシステム３２は、実行待ち行列３３を含み、実行待ちのジョブ４１、４２の順序を管理する。スケジューラ３１は、上述の方法に従って、動的および静的スケジューリングを行う。各ＰＥは、図３に示すように、ＣＰＵ５１およびメモリ５２を備え、ジョブ４１、４２を実行する。

【００２４】キューイングシステム３２からＰＥに投入されたジョブ４１、４２は、動的スケジューリングにより、他のＰＥに移動したり、凍結されて外部補助記憶装置２３に書き出されたりする。ここで、ジョブ４１は、複数のＰＥ上で実行される並列ジョブに対応し、ジョブ４２は、１つのＰＥ上で実行されるジョブに対応する。

【００２５】また、凍結されたジョブは、その後、外部補助記憶装置２３から読み出されて解凍され、動作を継続する。ここで、ジョブの解凍とは、凍結により外部補助記憶装置２３に保存されたジョブの情報を、ＰＥ上に復元する処理を表す。解凍先のＰＥは、凍結時の実行ＰＥまたは他のＰＥである。解凍先のＰＥが元のＰＥと異なる場合は、結果的にジョブが移動することになる。

【００２６】このように、並列計算機システムにおいて実行中のジョブの凍結や移動を行うことで、計算機資源の不要なアイドル状態等の非効率的な状態を改善し、システム本来の性能を引き出すことができる。また、凍結や移動により特定のＰＥ資源を解放することにより、システムの部分的な保守作業を動的に行うことができる。したがって、並列計算機システムをより効率的に利用することが可能になる。

【００２７】図２の並列計算機システムでは、以下の観点から切り分けた各階層においてスケジューリングを行い、ジョブを並べ替える。
（１）第１階層
システム立ち上げ時の逐次処理（システム立ち上げから待ち行列がジョブで飽和するまで）
（２）第２階層
待ち行列内におけるジョブの静的スケジューリング
（３）第３階層
待ち行列からＰＥへのジョブ投入時の動的スケジューリング（ＰＥと外部補助記憶装置の間の入出力）
（４）第４階層
ＰＥ上の実行中のジョブの動的スケジューリング
以下、図４から図１６までを参照しながら、第１～第４の各階層におけるスケジューリングについて詳細に説明する。

【００２８】図４は、第１階層におけるスケジューリングを示している。この階層では、システムが立ち上がった状態、すなわち、待ち行列３３内に後続のジョブが存在せず、計算機資源に空きがある状態を扱う。この場合、スケジューラは、ＦＩＦＯによりジョブを逐次投入するようなスケジューリングを行う。

【００２９】図４では、ＰＥ１およびＰＥ２上で並列ジョブ４１が実行されているときに、ジョブ４２がキューイングシステム内の待ち行列３３に投入される。しかし、ＰＥ３～ＰＥＰの資源が空いているので、ジョブ４２は直ちに実行される。このように、システムは、待ち行列３３に投入されるジョブを逐次実行可能な状態にある。

【００３０】次に、図５は、第２階層におけるスケジューリングを示している。この階層では、第１階層にて逐次処理を続けた後、空き資源が不足して待ち行列３３にジョブがたまっている状態を扱う。この場合、ＰＥ上のいずれかのジョブが終了し、次のジョブが実行可能になるまでに、スケジューラは、待ち行列３３内のジョブを並べ替えて、次に最も有効に投入可能なジョブを、待ち行列３３の先頭に持ってくるようにする。

【００３１】ここでは、スケジューラは、次のような優先順序で待ち行列３３内のジョブをソートする。このソート処理は、例えば、待ち行列３３に新たなジョブが投入される度に行われる。
（１）優先度（緊急度）の高い順にソート
（２）並列度の小さい順にソート
（３）メモリサイズの小さい順にソート
このようなソート処理によれば、まず、優先度によるジョブの分類・ソーティングが行われ、次に、同じ優先度

のジョブが、並列度／メモリサイズの小さい順にソートされる。言い換えれば、優先度が高く、かつ、小規模なジョブがなるべく早期に実行されるように、ジョブが並べ替えられる。

【００３２】この理由は、経験的に、並列度／メモリサイズの小さなジョブは比較的実行時間が短く、また、ジョブの凍結や移動の際にも、操作に必要な時間が短いからである。このように、実行時間が短く機動性の高いジョブを早期に実行することにより、それらのジョブを比較的早期に終了させることができる。

【００３３】次に、図６は、第３階層におけるスケジューリングを示している。この階層では、実際に待ち行列３３からＰＥにジョブを投入し、実行させるタイミングでスケジューリングが行われる。

【００３４】この場合、キューイングシステム３２は、現在実行中のジョブの情報を管理するジョブ情報テーブル６１を保持し、スケジューラは、ジョブの凍結や移動に備えて、次の優先順序でジョブ情報テーブル６１内のジョブをソートする。このソート処理は、例えば、各階層でスケジューリングが行われる度に行われる。

（１）優先度の高い順にソート
（２）並列度の小さい順にソート
（３）メモリサイズの大きい順にソート

図７は、ジョブ情報テーブル６１の例を示している。図７のジョブ情報テーブルには、現在実行中のジョブのジョブ名、識別情報（ＩＤ）、優先度、並列度、メモリサイズ、ジョブ投入時刻、アイドル時間等の情報が登録されている。このうち、ジョブ投入時刻は、ジョブの実行開始日時を表し、アイドル時間は、凍結処理等により実行が中断されていた時間を表す。

【００３５】例えば、第２階層でのスケジューリングが行われると、スケジューラは、待ち行列３３の先頭とジョブ情報テーブル６１の先頭のジョブを比較し、前者の優先度が後者のそれより高ければ、実行中のジョブを凍結して外部補助記憶装置２３に退避させ、資源の一部を空き状態にする。これにより、ジョブ情報テーブル６１のジョブが並べ替えられ、待ち行列３３の先頭のジョブが優先的に実行される。

【００３６】ここでは、スケジューラは、以下の手順により、退避させるジョブを選択する。この手順においては、ジョブの並列度とメモリサイズを合わせて、ジョブのサイズと呼ぶことにする。

【００３７】Ｐ１：ジョブ情報テーブル６１の最後尾から検索し、待ち行列３３の先頭のジョブが必要とするサイズを越えないような実行中ジョブのうち、最も大きなサイズのジョブを選択する。そのようなジョブが存在しなければ、ジョブ情報テーブル６１の最後尾のジョブを選択する。

【００３８】Ｐ２：待ち行列３３の先頭のジョブのサイズと、ジョブ情報テーブル６１から選択されたジョブのサイズの差を、次式により求め、それを必要なサイズとして用いて、Ｐ１およびＰ２の処理を繰り返す。

サイズの差＝待ち行列の先頭のジョブのサイズ
－ジョブ情報テーブルから選択されたジョブのサイズの和　　（１）

そして、サイズの差が０もしくは負になれば、この処理を終了する。

【００３９】Ｐ３：ジョブ情報テーブル６１の情報から、各ジョブの推定終了時刻と選択されたジョブの凍結処理に必要な時間を計算し、並べ替えオーバヘッドにより総実行コスト（総実行時間）が増加するか否かをチェックする。そして、総実行コストが増加する場合は、このスケジューリングを中止し、ＰＥ上のジョブが自然に終了するのを待つ。ジョブの実行コストの計算方法については、後述することにする。

【００４０】Ｐ４：総実行コストが増加しない場合、選択されたジョブと待ち行列３３の先頭のジョブを入れ替えるのが妥当であると判断し、選択されたジョブを凍結して、外部補助記憶装置２３に退避させる。これにより、待ち行列３３の先頭のジョブにための資源が確保される。

【００４１】Ｐ５：待ち行列３３の先頭のジョブをＰＥ上に投入し、それをＰＥに実行させる。図６では、ＰＥ１～ＰＥ３上のｊｏｂ＿ＺＺとｊｏｂ＿ＣＣの２つのジョブが選択され、外部補助記憶装置２３に退避させられる。そして、空き状態になった資源を利用して、待ち行列３３の先頭のジョブ２２が実行される。

【００４２】このようなスケジューリングによれば、待ち行列３３内の優先度の高いジョブを、より優先度の低い実行中ジョブに優先して実行することができる。また、ジョブ情報テーブル６１のジョブを優先度／並列度／メモリサイズによりソートしておくことで、退避に適したジョブを容易に選択することができ、最も効率の良い退避操作を行うことができる。

【００４３】次に、第４階層におけるスケジューリングについて説明する。この階層では、スケジューラは、ＰＥの負荷分散とシステムの保守を目的として、実行中のジョブと凍結により退避中のジョブのスケジューリングを行う。これらのジョブは、上述したジョブ情報テーブルに登録されている。ここでは、動作中のジョブを動作させたままでシステム内の別のＰＥに移動させるマイグレーションと、運用中におけるシステム構成の変更とを想定して、動的スケジューリングを行う。

【００４４】例えば、ＣＰＵ資源やメモリ資源を多く消費するような、処理の重いジョブがあるＰＥに集中した場合、資源に余裕のある他のＰＥにジョブを移動することで、各ＰＥの負荷を平均化することができ、システム

全体の処理時間を短縮することができる。

【００４５】図８は、システム運用中の負荷分布の例を示している。この例では、ＰＥ１～ＰＥ６の６個のＰＥが稼働しており、これらのＰＥがそれぞれ２個のジョブ４２を実行している。ジョブ４２の矩形内の数字は、そのジョブの負荷量を表し、ＰＥの矩形内の数字は、そのＰＥの負荷量を表す。ＰＥの負荷量は、そのＰＥ上のジョブの負荷量の総和に相当する。例えば、ＰＥ１上の２つのジョブ４２の負荷量は、それぞれ“１０”および“５”であり、ＰＥ１の負荷量は“１５”である。

【００４６】次に、時間の経過とともに＊印のジョブが終了すると、図９に示すような状態になる。この状態では、ＰＥ１およびＰＥ２の負荷が重く、ＰＥ５およびＰＥ６の負荷が軽い（負荷が“０”である）ため、全体としてＰＥ負荷がアンバランスになっている。

【００４７】そこで、マイグレーションを用いた動的スケジューリングにより、一部のＰＥ負荷を分散させることを考える。例えば、図９のＰＥ１およびＰＥ２上の負荷量“５”のジョブ４２を、それぞれＰＥ５およびＰＥ６上に移動させるようなスケジューリングを行えば、図１０に示すように、ＰＥ負荷が平均化される。

【００４８】また、システム構成の変更とは、例えば、稼働中のＰＥを停止したり、新たなＰＥを追加したりすることに対応する。特定のＰＥを保守や節電等の理由で停止したい場合、通常は、そのＰＥに関連するジョブを強制終了させるか、あるいは、これらのジョブが自然に終了するのを待たなければならない。しかし、マイグレーションを用いた動的スケジューリングを行えば、関連するジョブを最適な場所に移動させることができる。したがって、ジョブの実行を継続しながら、ＰＥを停止させることができ、システム構成の変更に対応して、効率的な運用が可能になる。

【００４９】例えば、図１０においてＰＥ５およびＰＥ６を停止したい場合、ＰＥ５およびＰＥ６上のジョブ４２を、それぞれＰＥ３およびＰＥ４上に移動させるようなスケジューリングを行えばよい。これにより、図１１に示すように、ＰＥ５およびＰＥ６を直ちに停止することができ、さらに、残りのＰＥの負荷が平均化される。

【００５０】ところで、ジョブの凍結やマイグレーションの際には、対象ジョブが使用しているすべての資源を解放することを前提に、スケジューラは、ジョブに関連する以下のデータをパッキングする。

（１）ジョブを構成するプロセスのユーザ空間（実行オブジェクトが置かれたメモリ空間、実行オブジェクトが参照するメモリ空間等）

（２）ジョブを構成するプロセスに関連するＯＳ内の制御表（プロセス管理テーブル等）

次に、凍結の場合は、パッキングしたデータをファイルとして外部補助記憶装置に書き出し、マイグレーションの場合は、パッキングしたデータを移動先のＰＥに転送して展開する。このように、ジョブに関連するすべてのデータをパッキングして退避／転送することで、ＰＥ上でジョブが占有していた資源が完全に解放される。

【００５１】ＵＮＩＸ（登録商標）等のシステムの場合、ジョブを構成する各プロセスには、プロセスを識別するためにシステム内でユニークに設定されたＩＤが付加されている。このため、凍結／解凍、あるいはマイグレーションの操作により、システム内でプロセスＩＤの矛盾（衝突）が発生する可能性がある。

【００５２】このプロセスＩＤの矛盾を回避するためには、例えば、特開平（「ジョブ実行システム」）または特開平１１－３３８７１９（「計算機システム」）に記載された技術を用いればよい。

【００５３】特開平１１－３３８７１９では、ジョブを構成するプロセスに対して、ジョブＩＤと、そのプロセスを実行するＰＥの仮想プロセッサＩＤと、そのＰＥ内のローカルプロセスＩＤとから生成されたプロセスＩＤが割り付けられる。したがって、ジョブが退避させられた後に、それを構成するプロセスと同じローカルプロセスＩＤを持つプロセスが生成されても、プロセスＩＤの衝突は生じない。

【００５４】ここで、第３および第４階層のスケジューリングにおいて用いられるジョブの実行コストおよび移動コストの計算方法について説明しておく。ジョブの実行コストは、そのジョブの実行時間に対応し、一般に、コンパイラがジョブのコンパイル時に実行コストをオブジェクトに埋め込まない限り、実行段階でそのジョブの実行コストを正確に算出することはできない。しかし、キューイングシステムは、ジョブを割り当てる際の指標とするために、ジョブの実行メモリサイズを管理しているので、この値を参考にして実行コストを推定することができる。

【００５５】ここでは、以下のような相関関係を利用して、実行コストを算出する。

（＃１）一般に、ジョブの実行コストは、使用するメモリサイズに比例する。

（＃２）同一ジョブの実行には、常に同一の実行コストがかかる。

（＃３）１つのＰＥに複数のジョブが存在する場合、ＰＥ内での逐次実行（プロセススイッチング）によるオーバヘッドは、ＰＥ内に存在するジョブの使用するメモリ総量に比例する。

【００５６】これらの条件をすべて定量化して、すべてのジョブの総実行コストを算出する。まず、（＃１）の条件を基準にして、ジョブのメモリサイズの値Mem_{job}［ｂｙｔｅ］と実行コストＴの関係を表すと、次式のようになる。

$$T = Mem_{job} \times \sigma \quad (2)$$

ここで、比例係数σは、ジョブのメモリサイズと実行コストの関連を定義するパラメータであり、ジョブの実際の実行時間から決められる。

【００５７】次に、（＃２）の条件を基準にしてσを決定し、各ジョブのメモリサイズと実行コストの関係を得る。ここでは、過去のジョブ実行コスト履歴として、実

$$T = (1/N)\ \Sigma t_j$$

ここで、Σはｊに関する総和を表す。（２）および（３）式より、このジョブの比例係数σは次のように決

$$\sigma = (1/N \times Mem_{job})\ \Sigma t_j$$

ただし、初めて実行されるジョブの場合には、σとして適当な値を設定しておき、それを随時更新しながらシステムの運用を続けることで、σの精度を向上させる。

【００５８】また、ジョブの形態により、ジョブ自身のサイズは等しくても、入力されるデータファイルのサイ

$$\delta = (1/N)\ \Sigma\ (T - t_j)^2$$

このδが一定しきい値以上であれば、それらの実行時間ｔｊをデータベースの更新に用いず、δが一定しきい値未満であれば、実行時間ｔｊを用いてデータベースを更新する。

【００５９】以上のコスト計算によればジョブの最短実行コストが得られるが、ＰＥ上で複数のジョブが実行される場合、実際には、プロセスのスイッチングに伴うメモリの切り替えによるオーバヘッドが発生する。そこ

$$Mt = \Sigma Mem_{job}$$

ここで、ΣはＰＥ内のすべてのジョブに関する総和を表す。このＭｔを用いて、オーバヘッド時間Ｔｏは、次式

$$To = Mt \times \lambda$$

ここで、比例係数λは、システムの設定により決められる定数である。この場合、各ジョブの実行コストは、

$$Cexec = T + To$$

スケジューリングの計算は、ジョブ実行の裏で並列して行われるため、その計算に要するスケジューリングオーバヘッドは、ジョブ実行コストにより隠蔽される。

【００６２】また、ジョブの投入操作や、凍結／解凍またはマイグレーションによる移動操作が行われた場合、（＃３）のオーバヘッドによりジョブの実行時間が変化する。そこで、ジョブの投入時刻と現在時刻から経過時間を算出し、残りの実行コストを補正する。

【００６３】ところで、ジョブの移動処理としては、次の２つの処理が考えられる。

（ａ）凍結されて外部補助記憶装置に退避したジョブを、凍結前のＰＥとは別のＰＥ上で解凍する。

（ｂ）マイグレーションにより、パッキングしたデータ

$$Cfreeze = Cdefrost = Mem_{job} / IO$$

また、ネットワークの通信性能（データ転送性能）をＤＴ［ｂｙｔｅ／ｓｅｃ］とすると、マイグレーションに

$$Cmigrate = Mem_{job} / DT$$

例えば、ＶＰＰシステムでは、ＩＯ＝６０［Ｍｂｙｔｅ／ｓｅｃ］であり、ＤＴ＝３２［Ｇｂｙｔｅ／ｓｅｃ］

際のジョブの実行コストのデータベースを作成し、同一ジョブがＮ回実行されたとき、それらの実行時間の平均をＴとする。Ｎ回の各々の実行時間をｔｊ（ｊ＝１，．．．，Ｎ）とすると、それらの平均は次式により求められる。

（３）

められる。

（４）

ズによって実行コストが変化する場合がある。この場合、実行コストのデータベースの更新に用いるデータの選択基準として、次式により実行時間ｔｊの分散値δを求め、その値に応じてデータの取捨選択を行う。

（５）

で、（＃３）の条件を基準にして、このオーバヘッドを計算する。

【００６０】システムの設定にもよるが、一般に、オーバヘッド時間Ｔｏは、ジョブのメモリ使用量に比例して増大する。そこで、ある時点におけるあるＰＥ内のジョブのメモリ使用量をMem_{job}とすると、総メモリ使用量Ｍｔは次式により与えられる。

（６）

により近似することができる。

（７）

（２）式のＴから次式のＣexecに置き換えられる。

【００６１】

（８）

をネットワーク経由で転送する。

【００６４】上述したように、ジョブのパッキングデータは、メモリ空間とＯＳの制御表から構成されている。このうち、ＯＳの制御表はジョブによらずに一定であるので、パッキングに要するオーバヘッドは、ジョブのメモリ空間のみに依存する。

【００６５】（ａ）および（ｂ）の処理における転送媒体は、それぞれ、外部補助記憶装置およびネットワークである。そこで、外部補助記憶装置のデータ入出力性能をＩＯ［ｂｙｔｅ／ｓｅｃ］とすると、凍結に要するコストＣfreezeと解凍に要するコストＣdefrost は、次式で与えられる。

（９）

要するコストは、次式で与えられる。

（１０）

である。（９）、（１０）式のコストを移動コストとして用いて、動的スケジューリングが行われる。

【００６６】また、ジョブの投入時刻から既に時間ｔが経過している場合、そのジョブの残り時間はＣexec－ｔとなる。したがって、Ｃexec－ｔ＜ＣfreezeまたはＣexec－ｔ＜Ｃmigrate となるような場合は、そのジョブの移動コストが残り時間より大きいので、動的スケジューリングは行われない。

【００６７】以上のコスト計算は、第３および第４階層のスケジューリングにおいて行われるものであるが、これを第２階層のスケジューリングに適用することもできる。この場合、例えば、（３）式の実行コストを用いて待ち行列内のジョブを評価し、実行コストの小さい順にジョブをソートする。

【００６８】第４階層のスケジューリングでは、スケジューラは、ジョブの凍結や移動を考慮して、ＰＥ上のジョブを再配置する。このとき、再配置に要するオーバヘッドも考慮することで、システム運用の効率化を図る。ここでは、効率化のために、以下のような操作を行う。

（１）過去のジョブ実行時の実績データをジョブ実行履歴として蓄積しておき、過去に実行された同一ジョブの実行時間の統計をとる。

（２）ジョブの凍結や移動の操作コストを、操作対象のジョブのメモリサイズから計算して、再配置に要するオーバヘッド時間を見積もる。

（３）実行時間の統計情報とオーバヘッド時間を比較し、ジョブの再配置が完了する前に終了すると予測されるジョブについては、再配置を行わない。

（４）再配置を行わないジョブが占有している計算機資源については、スケジューリングの際にマスクしておく。

【００６９】このスケジューリングは、例えば、以下のような場合を契機として行われる。

（１）システム保守やＰＥの故障により、ＰＥ構成が変化する場合

（２）ＰＥの負荷状況を監視し、ＰＥ間で負荷状況がアンバランスな場合

（３）一定のインターバルタイムを設定し、定期的にスケジューリングを行う場合

（２）の負荷状況の監視は、（３）のインターバルタイムに基づいて定期的に行われる他、ジョブが終了する毎にも行われる。また、負荷状況は、以下のような方法で判定される。

【００７０】ｐ番目のＰＥの総メモリ使用量をM_{PEp}とすると、Ｐ個のＰＥからなるシステムの平均負荷Ｌｏａｄは、次式で与えられる。

$$Load=(\lambda 1/P)\Sigma M_{PEp} \quad (11)$$

ここで、比例係数λ１は、システムの設定により決められる定数であり、ΣはＰ個のＰＥについての総和を表す。このＬｏａｄを用いて、各ＰＥの負荷の分散値δ_{load}は、次式により求められる。

$$\delta_{load}=(1/P)\Sigma(Load-\lambda 1 M_{PEp})^2 \quad (12)$$

システムは、この分散値δ_{load}を、あらかじめ決められたしきい値δ０と比較し、δ_{load}がδ０より大きい場合に、負荷状況がアンバランスであると判定する。ここでは、メモリ使用量に関する負荷を監視しているが、代わりに、ＣＰＵやデータ入出力に関する負荷を監視してもよい。

【００７１】図１２は、上述のようなスケジューリングの契機を示すフローチャートである。並列計算機システムは、通常の運用動作を行い（ステップＳ１）、いずれかのジョブが終了したか否かをチェックする（ステップＳ２）。いずれのジョブも終了していなければ、次に、現在の時刻が定期的なスケジューリングポイントに対応するか否かをチェックする（ステップＳ３）。

【００７２】現在の時刻がスケジューリングポイントに対応すれば、次に、各ジョブのメモリサイズ７１を用いて、各ＰＥの負荷の分散値を計算し（ステップＳ４）、負荷状況がアンバランスか否かを判定する（ステップＳ５）。そして、負荷状況がアンバランスであれば、スケジューリングを実施し（ステップＳ６）、通常の運用動作に戻る。

【００７３】ステップＳ２においていずれかのジョブが終了した場合は、そのままステップＳ４以降の処理を行う。また、ステップＳ３において現在の時刻がスケジューリングポイントに対応しない場合、および、ステップＳ５において負荷のバランスが取れている場合は、スケジューリングを行わずに通常の運用動作に戻る。

【００７４】図１３は、図１２のステップＳ６で行われるスケジューリング処理のフローチャートである。スケジューラは、まず、ジョブ情報テーブル６１に登録されているジョブを、以下の優先順序でソートする（ステップＳ１１）。

（１）優先度（緊急度）の高い順にソート

（２）メモリサイズの大きい順にソート

次に、ジョブ情報テーブル６１から、ＰＥ上に再配置されたジョブを含む再配置リスト８１を生成する（ステップＳ１２）。次に、各ジョブのメモリサイズ７１、各ジョブの実行経過時間７２、各ジョブの実行コスト履歴７３、外部補助記憶装置のデータ入出力性能７４、およびネットワークのデータ転送性能７５に基づいて、上述した計算方法により、各ジョブの残り時間を推定し、移動コストを求める（ステップＳ１３）。そして、残り時間と移動コストを比較して、各ジョブが移動可能か否かを判定する。

【００７５】こうして、再配置リスト８１に登録されたジョブは、移動しないジョブ８２、通常ジョブ８３、移動するジョブ８４、および凍結対象ジョブ８５に分類される。このうち、通常ジョブ８３は、移動しても、しなくてもよいジョブに対応し、凍結対象ジョブ８５は、ジ

ョブの凍結処理８６により外部補助記憶装置に退避させられる。

【００７６】次に、再配置リスト８１のジョブを分類結果に基づいてソートし（ステップＳ１４）、再配置リスト８１を最適化する（ステップＳ１５）。そして、最適化された再配置リスト８１に基づいてジョブの再割当を行い（ステップＳ１６）、処理を終了する。

【００７７】図１４は、図１３のステップＳ１２で行われる再配置リスト生成処理のフローチャートである。スケジューラは、まず、システム上のＰＥから保守等のために停止するＰＥ（または、ジョブを割り当てたくないＰＥ）を除外して、残りのＰＥを含む運用ＰＥリストを生成する（ステップＳ２１）。

【００７８】次に、ソート後のジョブ情報テーブルのジョブのうち、除外されたＰＥに割り付けられているジョブに、「移動するジョブ」のフラグを立てる（ステップＳ２２）。このフラグは、強制的な移動対象のジョブであることを表し、このフラグが立っているジョブは、マイグレーション処理の対象となる。また、ＰＥの停止により強制終了させられたジョブは、待ち行列に追加される。

【００７９】次に、ソート後のジョブ情報テーブルの先頭のジョブから順に、運用ＰＥリストのＰＥに対して、計算機資源の許す限り、仮定的に割り当てていき、ＰＥに割り当てられたジョブを再配置リストに登録する（ステップＳ２３）。そして、ＰＥに割り当てられなかったジョブを凍結対象ジョブに決定して（ステップＳ２４）、処理を終了する。

【００８０】図１３のステップＳ１３では、スケジューラは、各ジョブの実行コストＣexecと経過時間ｔから、終了するまでに要する残り時間Ｃexec－ｔを求め、凍結対象ジョブの凍結コストＣfreezeを算出する。そして、ＣfreezeがＣexec－ｔより大きい場合は、そのジョブを凍結対象ジョブから除外し、このジョブの終了に合わせて、再スケジューリングを行う時刻を修正する。

【００８１】次に、システム上で実行されるジョブのマイグレーションコストＣmigrate を算出し、ＣmigrateがＣexec－ｔより大きいジョブに、「移動しないジョブ」のフラグを立てる。これにより、移動が完了する前に終了すると予測されるジョブが再配置対象から除外され、そのジョブが使用している資源がマスクされる。

【００８２】また、図１３のステップＳ１４では、スケジューラは、再配置リストのジョブを、以下の優先順序でソートする。

（１）「移動しないジョブ」のフラグが立っているグループ、フラグなしのグループ（通常ジョブ）、および「移動するジョブ」のフラグが立っているグループが、それぞれ、リストの前部、中間部、および後部にくるようにソートする。

（２）各グループ内のジョブをメモリサイズの大きい順にソートする。

【００８３】図１５は、図１３のステップＳ１５で行われる最適化処理のフローチャートである。スケジューラは、まず、ソート後の再配置リストにおいて、通常ジョブのグループの先頭のジョブの順位を変数ｎに代入し（ステップＳ３１）、１番目～ｎ－１番目のジョブを移動しないジョブとみなして、元のＰＥに割り当てる（ステップＳ３２）。

【００８４】次に、ｎ番目以降のジョブを再配置対象のジョブとみなして、空きＰＥに割り当て、新たな再配置リストを生成する（ステップＳ３３）。このとき、移動しないジョブにより既に資源を占有されているＰＥに配慮して、負荷（メモリ使用量）は均等になるように、ジョブを配置する。具体的には、資源の余裕が大きいＰＥ群に対して、メモリサイズの大きなジョブから順番に割り当てる。ここでは、ジョブがメモリサイズの大きい順にソートされているので、ソート順にジョブをＰＥに割り当てればよい。

【００８５】次に、再配置により移動するジョブのマイグレーションコストＣmigrate を算出し、それらの最大値Ｃmax を求める（ステップＳ３４）。次に、ｎ＝ｎ＋１とおき（ステップＳ３５）、ｎ番目のジョブに「移動するジョブ」のフラグが立っているか否かをチェックする（ステップＳ３６）。そして、ｎ番目のジョブが移動するジョブでなければ、ステップＳ３２以降の処理を繰り返す。こうして、複数の新たな再配置リストが生成される。

【００８６】ステップＳ３６において、「移動するジョブ」のフラグが現れれば、次に、得られた再配置リストのうち、Ｃmax の値が最も小さいものを、最適な再配置リストとして選択し（ステップＳ３７）、処理を終了する。

【００８７】図１６は、図１５のステップＳ３３で行われる割当処理の例を示すフローチャートである。ここでは、運用ＰＥリストに登録されたＰＥの数をＰとしている。スケジューラは、まず、ＰＥの番号を表す変数ｐに１を代入し（ステップＳ４１）、ｎ番目のジョブの並列度を変数ａに代入する（ステップＳ４２）。

【００８８】次に、ｐ＋ａ－１とＰを比較する（ステップＳ４３）。ｐ＋ａ－１がＰ以下であれば、ｎ番目のジョブにｐ番目～ｐ＋ａ－１番目のＰＥを割り当て（ステップＳ４４）、ｐ＝ｐ＋ａとおく（ステップＳ４５）。

【００８９】次に、ｎ＝ｎ＋１とおいて（ステップＳ４６）、ｎ番目のジョブが存在するか否かをチェックする（ステップＳ４７）。ｎ番目のジョブが存在すれば、そのジョブについて、ステップＳ４２以降の処理を繰り返す。

【００９０】また、ステップＳ４３においてｐ＋ａ－１がＰを越えれば、ｎ番目のジョブに、１番目～ｐ＋ａ－１－Ｐ番目のＰＥとｐ番目～Ｐ番目のＰＥを割り当る

（ステップＳ４８）。そして、ｐ＝ｐ＋ａ－Ｐとおいて（ステップＳ４９）、ステップＳ４６以降の処理を行う。こうして、ステップＳ４７においてｎ番目のジョブがなくなれば、処理を終了する。

【００９１】以上説明したように、本実施形態のスケジューリングには、以下のような特徴がある。

（１）静的および動的な側面から総合的なスケジューリングを行う。

（２）過去の実行情報を蓄積することにより、確定不可能なジョブの実行時間を予測し、スケジューリングにフィードバックする。

（３）ジョブの凍結による外部補助記憶装置への退避と、ＰＥ間でのジョブの移動等を組み合わせて、より高度なスケジューリングを行う。

（４）ＰＥの停止のような、システム全体のオペレーションを考慮したスケジューリングを行う。

【００９２】このようなスケジューリングを行うことで、計算機資源を有効に利用することができる。図２のコンピュータ２１は、例えば、図１７に示すような情報処理装置に対応する。図１７の情報処理装置は、ＣＰＵ（中央処理装置）９１、メモリ９２、入力装置９３、出力装置９４、外部記憶装置９５、媒体駆動装置９６、およびネットワーク接続装置９７を備え、それらはバス９８により互いに接続されている。

【００９３】メモリ９２は、例えば、ＲＯＭ（read only memory）、ＲＡＭ（random access memory）等を含み、処理に用いられるプログラムとデータを格納する。ＣＰＵ９１は、メモリ９２を利用してプログラムを実行することにより、必要な処理を行う。例えば、図２のスケジューラ３１およびキューイングシステム３２は、プログラムとしてメモリ９２に格納される。

【００９４】入力装置９３は、例えば、キーボード、ポインティングデバイス、タッチパネル等であり、ユーザからの指示や情報の入力に用いられる。出力装置９４は、例えば、ディスプレイ、プリンタ、スピーカ等であり、ユーザへの問い合わせや処理結果の出力に用いられる。

【００９５】外部記憶装置９５は、例えば、磁気ディスク装置、光ディスク装置、光磁気ディスク（magneto-optical disk）装置、テープ装置等である。情報処理装置は、この外部記憶装置９５に、上述のプログラムとデータを保存しておき、必要に応じて、それらをメモリ９２にロードして使用する。

【００９６】媒体駆動装置９６は、可搬記録媒体９９を駆動し、その記録内容にアクセスする。可搬記録媒体９９としては、メモリカード、フロッピー（登録商標）ディスク、ＣＤ－ＲＯＭ（compact disk read only memory）、光ディスク、光磁気ディスク等、任意のコンピュータ読み取り可能な記録媒体が用いられる。ユーザは、この可搬記録媒体９９に上述のプログラムとデータを格納しておき、必要に応じて、それらをメモリ９２にロードして使用する。

【００９７】ネットワーク接続装置９７は、ネットワーク２２に接続され、ＰＥとの通信に伴うデータ変換を行う。また、ネットワーク接続装置９７は、他の任意の通信ネットワークに接続することもできる。この場合、情報処理装置は、上述のプログラムとデータをネットワーク接続装置９７を介して他の装置から受け取り、必要に応じて、それらをメモリ９２にロードして使用する。

【００９８】図１８は、図１７の情報処理装置にプログラムとデータを供給することのできるコンピュータ読み取り可能な記録媒体を示している。可搬記録媒体９９や外部のデータベース１００に保存されたプログラムとデータは、メモリ９２にロードされる。そして、ＣＰＵ９１は、そのデータを用いてそのプログラムを実行し、必要な処理を行う。

（付記１）　複数のプロセッサエレメントを有する並列計算機システムのジョブスケジューリングを行うスケジューリング装置であって、プロセッサエレメントで実行中のあるジョブを他のプロセッサエレメントに移動するべきか否かを判定する判定手段と、前記ジョブが前記他のプロセッサエレメントに移動するべきであると判定されたとき、該ジョブのマイグレーション処理が行われるように、前記複数のプロセッサエレメントに対する実行中のジョブの割当を行う割当手段とを備えることを特徴とするスケジューリング装置。

（付記２）　前記複数のプロセッサエレメントの負荷状況を監視する監視手段をさらに備え、前記複数のプロセッサエレメント間で負荷分布がアンバランスになったとき、前記割当手段は、前記実行中のジョブの割当を行うことを特徴とする付記１記載のスケジューリング装置。

（付記３）　前記判定手段は、前記実行中のジョブの情報を含むジョブ情報テーブルを生成し、該ジョブ情報テーブルのジョブのうち、移動するジョブを決定して、前記複数のプロセッサエレメント上に再配置されたジョブの情報を含む再配置リストを生成し、前記割当手段は、該再配置リストに基づいて、該実行中のジョブの割当を行うことを特徴とする付記１記載のスケジューリング装置。

（付記４）　前記判定手段は、前記ジョブのマイグレーション処理に要するコストを計算し、計算されたコストに基づいて、前記ジョブを前記他のプロセッサエレメントにマイグレートするべきか否かを判定することを特徴とする付記１記載のスケジューリング装置。

（付記５）　前記判定手段は、前記ジョブの実行履歴情報に基づいて、該ジョブの実行コストを推定し、推定された実行コストを用いて、前記ジョブを前記他のプロセッサエレメントにマイグレートするべきか否かを判定することを特徴とする付記１記載のスケジューリング装置。

（付記６）　複数のプロセッサエレメントを有する並列計算機システムのジョブスケジューリングを行うスケジューリング装置であって、静止状態にあるジョブのスケジューリングを行う静的スケジューリング手段と、動作状態にあるジョブのスケジューリングを行う動的スケジューリング手段とを備えることを特徴とするスケジューリング装置。

（付記７）　複数のプロセッサエレメントを有する並列計算機システムのジョブスケジューリングを行うスケジューリング装置であって、前記並列計算機システムの構成が固定されている場合のスケジューリングを行う静的スケジューリング手段と、前記並列計算機システムの構成が変更可能な場合のスケジューリングを行う動的スケジューリング手段とを備えることを特徴とするスケジューリング装置。

（付記８）　複数のプロセッサエレメントを有する並列計算機システムであって、プロセッサエレメントで実行中のジョブを他のプロセッサエレメントに移動するべきか否かを判定する判定手段と、前記ジョブが前記他のプロセッサエレメントに移動するべきであると判定されたとき、該ジョブのマイグレーション処理が行われるように、実行中のジョブのスケジューリングを行うスケジューリング手段とを備えることを特徴とする並列計算機システム。

（付記９）　複数のプロセッサエレメントを有する並列計算機システムのジョブスケジューリングを行うコンピュータのためのプログラムを記録した記録媒体であって、前記プログラムは、プロセッサエレメントで実行中のジョブを他のプロセッサエレメントに移動するべきか否かを判定し、前記ジョブが前記他のプロセッサエレメントに移動するべきであると判定されたとき、該ジョブのマイグレーション処理が行われるように、前記複数のプロセッサエレメントに対する実行中のジョブの割当を行う処理を前記コンピュータに実行させることを特徴とするコンピュータ読み取り可能な記録媒体。

（付記１０）　複数のプロセッサエレメントを有する並列計算機システムのジョブスケジューリングを行うコンピュータのためのプログラムを記録した記録媒体であって、前記プログラムは、静的スケジューリングと動的スケジューリングを組み合わせたスケジューリング処理を、前記コンピュータに実行させることを特徴とするコンピュータ読み取り可能な記録媒体。

【００９９】

【実施例】次に、図１９から図２１までを参照しながら、上述した並列計算機システムの動作シミュレーションの結果について説明する。シミュレーション対象のモデルの詳細は、以下の通りである。

（１）ＰＥの数：５，１０，５０台

（２）ハードウェア：

・外部補助記憶装置のデータ入出力性能　６０［Ｍｂｙｔｅ／ｓｅｃ］

・通信装置のデータ転送性能　３２［Ｇｂｙｔｅ／ｓｅｃ］

・各ＰＥのメモリ容量　１０［Ｇｂｙｔｅ］

（３）ソフトウェア：１ＰＥ内４多重並列実行可能ＯＳ、キューイングシステム

（４）投入ジョブ：ポワソン分布に基づくランダムな特性を持った、以下のようなジョブ群

・ジョブの種類　５０種類

・並列度　１～１０

・メモリサイズ　１～１０Ｇ

・同一ジョブでの実行コストのばらつき　任意

・ジョブの到着時刻　任意

・投入ジョブ数　１０００

・ＰＥ構成の変更　任意の時刻に１０％のＰＥを停止し、任意の時刻に元の構成に戻す。

【０１００】以上のような同一のジョブ群に対して、以下の３種類のスケジューリングを適用するシミュレーションを行った。

（１）ＦＩＦＯによる逐次実行

（２）システムの空き資源を考慮した、待ち行列内のスケジューリング

（３）上述した実施形態の階層的スケジューリング（負荷の分散値のしきい値$\delta 0$として３種類の値を使用）

なお、（２）式のσの初期値は１［ｓｅｃ／ｂｙｔｅ］とし、（５）式の分散値δのしきい値は１２５０［sec^2］とし、（７）式の比例係数λは０．５［ｓｅｃ／ｂｙｔｅ］とした。ここでは、シミュレーションにおける時間の単位としてｓｅｃ（秒）を用いているが、これをクロック数等の他の単位に置き換えても、同様の結果が得られる。

【０１０１】また、第４階層のスケジューリングを行う契機を規定する（１２）式の分散値δ_{load}のしきい値$\delta 0$をパラメタとして、$\delta 0$＝２５，５０，７５の３種類の値を用いた。この値が小さいほど、スケジューリングが頻繁に行われることを意味する。

【０１０２】また、（１）および（２）のスケジューリング方法においてＰＥの停止を伴う場合、動作中のジョブの実行時間を監視し、ジョブの再投入を行う場合のコストと、ジョブの終了待ちを行う場合のコストを比較し、より有効な操作を選択している。

【０１０３】以上のようなシミュレーションにより、図１９、２０、および２１のような結果が得られた。図１９、２０、および２１のシミュレーション結果は、それぞれ５、１０、および５０台のＰＥからなる並列計算機システムのシミュレーションを表している。

【０１０４】ここで、総実行時間は、上述のジョブ群を実行するのに要した時間を表し、性能比は、（１）のスケジューリング方法の総実行時間を１としたときの他のスケジューリング方法の総実行時間の割合を表し、メモ

リ使用率は、ジョブ群が使用したメモリの割合を表す。これらの測定結果および実行状態から、以下のような傾向が読み取れた。

（１）システムが大規模になる（ＰＥの数が増加する）につれて、スケジューリング方法の違いによる性能差が小さくなる。

（２）第４階層のスケジューリングにおいて、ＰＥの数が少ない場合は、しきい値δ０を小さな値に設定して、スケジューリング頻度を高くすると効率がよい。また、ＰＥの数が多い場合は、しきい値δ０を大きな値に設定して、スケジューリング頻度を低くすると効率がよい。

（３）運用中にＰＥが停止した場合、（１）および（２）のスケジューリング方法では、ジョブの再投入または終了待ちによる処理の遅延が目立った。言い換えれば、ジョブの移動による継続運用が有効であることが分かった。

【０１０５】以上のような傾向から、（３）の階層的スケジューリングが有効であることが分かる。

【０１０６】

【発明の効果】本発明によれば、並列計算機システムにおいて、動作中のジョブの凍結や移動を考慮した動的スケジューリングを行うことで、ジョブ実行に伴うスケジューリング誤差を削減し、計算機資源の利用効率を向上させることができる。

【図面の簡単な説明】

【図１】本発明のスケジューリング装置の原理図である。

【図２】並列計算機システムの構成図である。

【図３】ＰＥの構成図である。

【図４】第１のスケジューリングを示す図である。

【図５】第２のスケジューリングを示す図である。

【図６】第３のスケジューリングを示す図である。

【図７】ジョブ情報テーブルを示す図である。

【図８】第１の負荷分布を示す図である。

【図９】第２の負荷分布を示す図である。

【図１０】第３の負荷分布を示す図である。

【図１１】第４の負荷分布を示す図である。

【図１２】第４のスケジューリングの契機を示すフローチャートである。

【図１３】スケジューリング処理のフローチャートである。

【図１４】再配置リスト生成処理のフローチャートである。

【図１５】最適化処理のフローチャートである。

【図１６】割当処理のフローチャートである。

【図１７】情報処理装置の構成図である。

【図１８】記録媒体を示す図である。

【図１９】第１のシミュレーション結果を示す図である。

【図２０】第２のシミュレーション結果を示す図である。

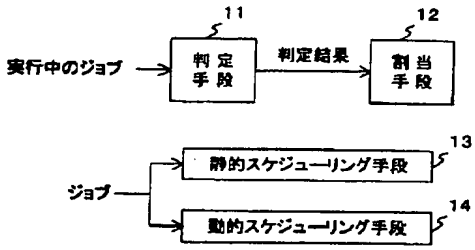
【図２１】第３のシミュレーション結果を示す図である。

【符号の説明】

１１　判定手段
１２　割当手段
１３　静的スケジューリング手段
１４　動的スケジューリング手段
２１　コンピュータ
２２　ネットワーク
２３　外部補助記憶装置
３１　スケジューラ
３２　キューイングシステム
３３　待ち行列
４１　並列ジョブ
４２　ジョブ
５１、９１　ＣＰＵ
５２、９２　メモリ
６１　ジョブ情報テーブル
７１　ジョブのメモリサイズ
７２　ジョブの実行経過時間
７３　ジョブの実行コスト履歴
７４　データ入出力性能
７５　データ転送性能
８１　再配置リスト
８２　移動しないジョブ
８３　通常ジョブ
８４　移動するジョブ
８５　凍結対象ジョブ
８６　ジョブの凍結処理
９３　入力装置
９４　出力装置
９５　外部記憶装置
９６　媒体駆動装置
９７　ネットワーク接続装置
９８　バス
９９　可搬記録媒体
１００　データベース

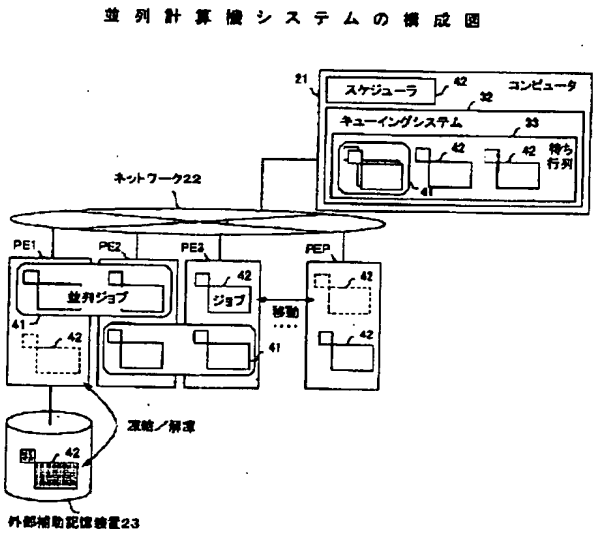
【図１】

本発明の原理図



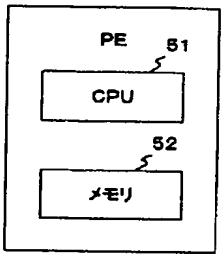
【図２】

並列計算機システムの構成図



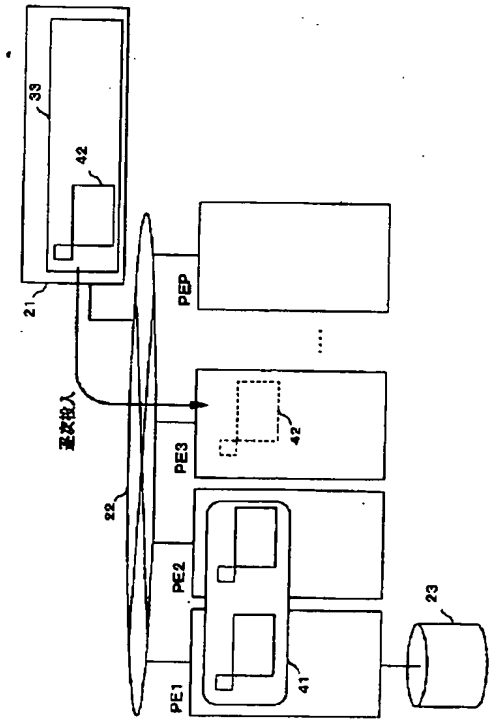
【図３】

ＰＥの構成図



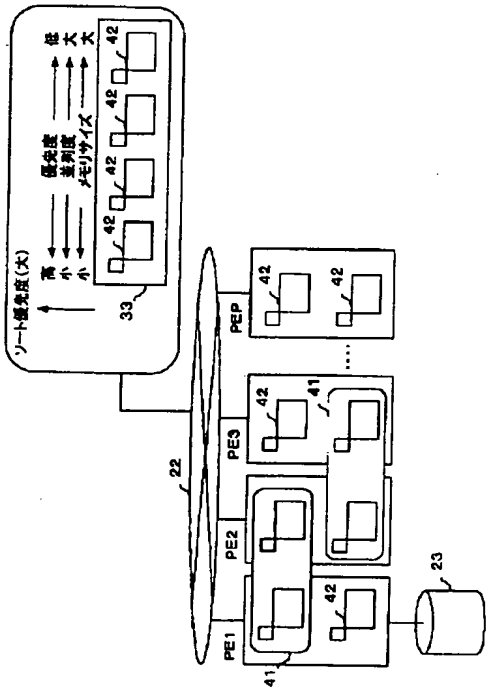
【図４】

第１のスケジューリングを示す図



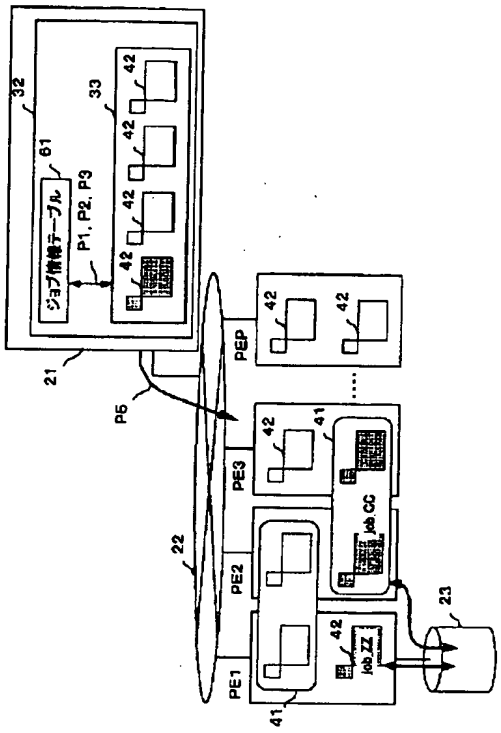
【図５】

第２のスケジューリングを示す図



【図６】

第３のスケジューリングを示す図



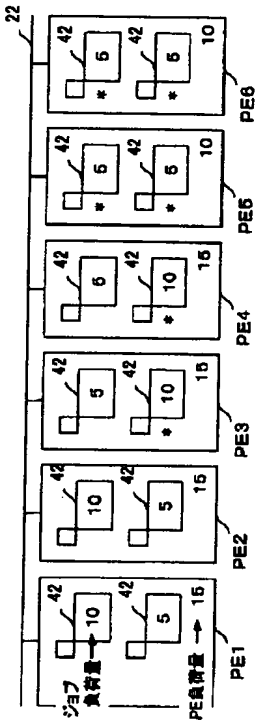
【図７】

ジョブ情報テーブルを示す図

ジョブ名	ジョブID	優先度	並列度	メモリサイズ	ジョブ投入時刻	アイドル時間
job_XX	105	1	6	100	990802083000	0
job_AA	110	1	8	100	990802090000	0
job_BB	107	1	6	50	990802084500	0
job_YY	99	2	8	100	990802080000	0
…						
job_ZZ	100	10	1	50	990802081000	5
job_CC	123	10	2	75	990802081500	10

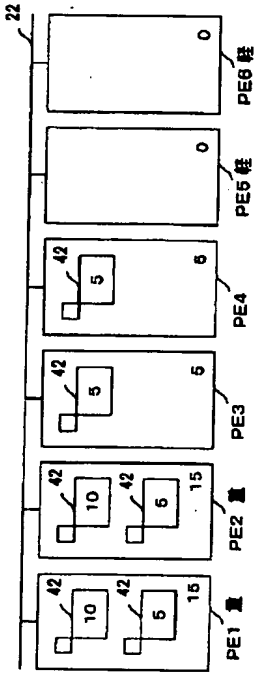
【図８】

第1の負荷分布を示す図



【図９】

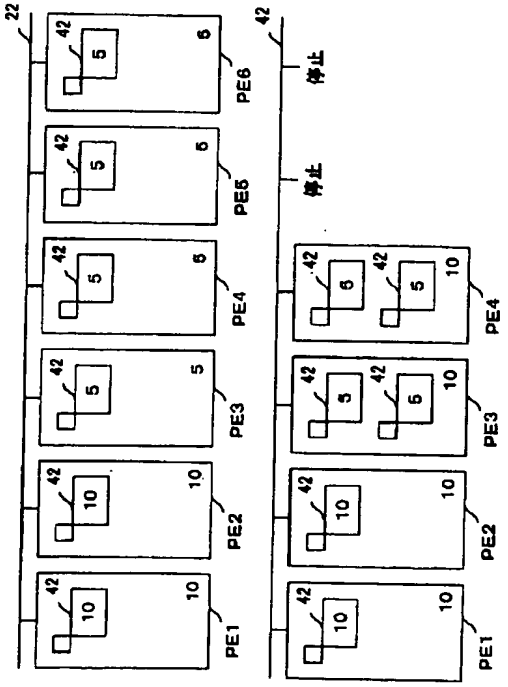
第2の負荷分布を示す図



【図１０】

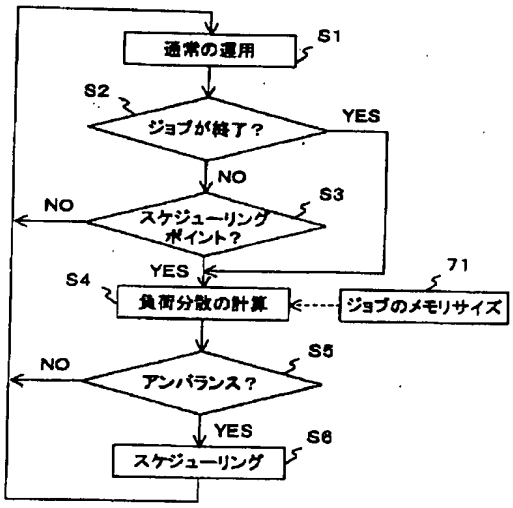
【図１１】

第3の負荷分布を示す図　第4の負荷分布を示す図



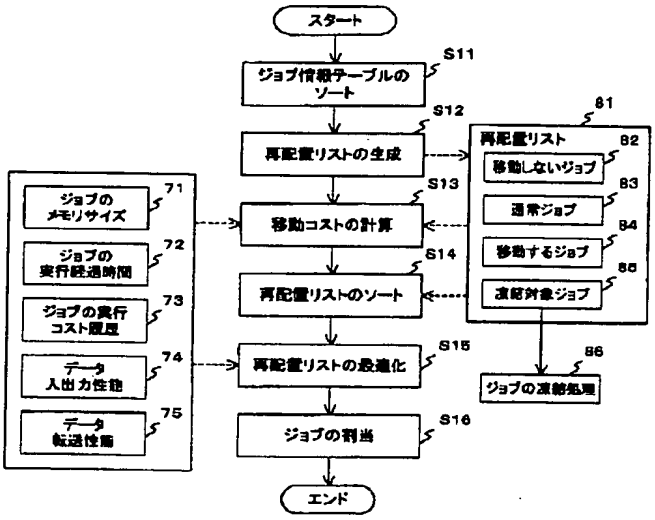
【図１２】

第4のスケジューリングの契機を示すフローチャート



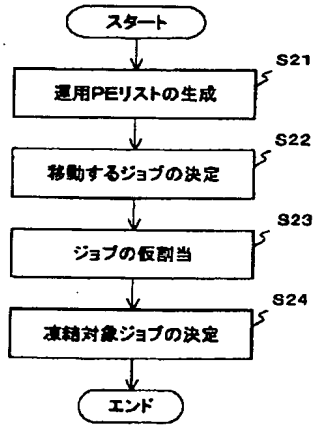
【図１３】

スケジューリング処理のフローチャート



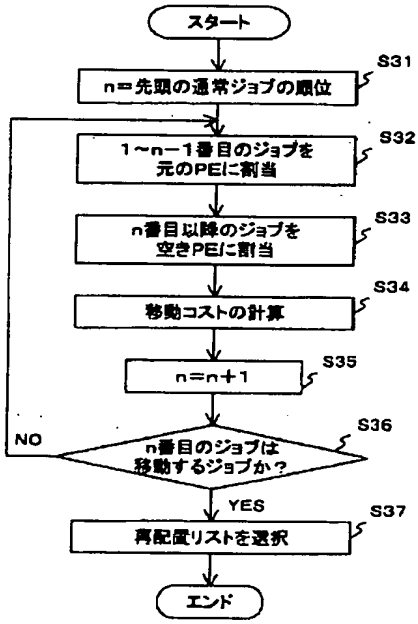
【図１４】

再配置リスト生成処理のフローチャート



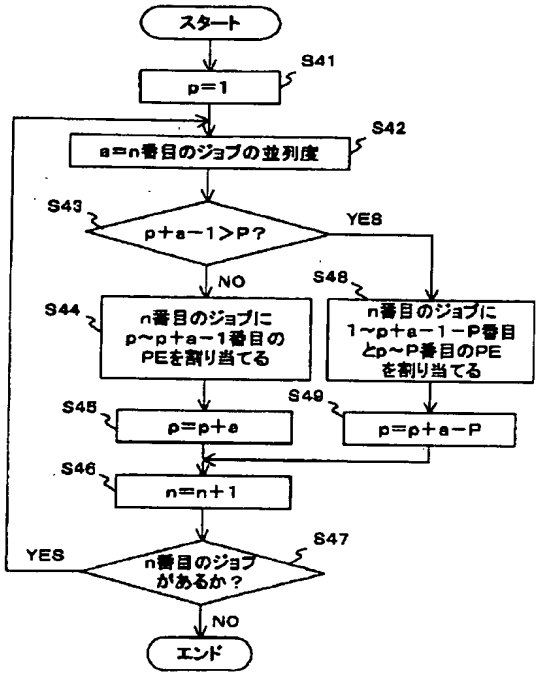
【図１５】

最適化処理のフローチャート



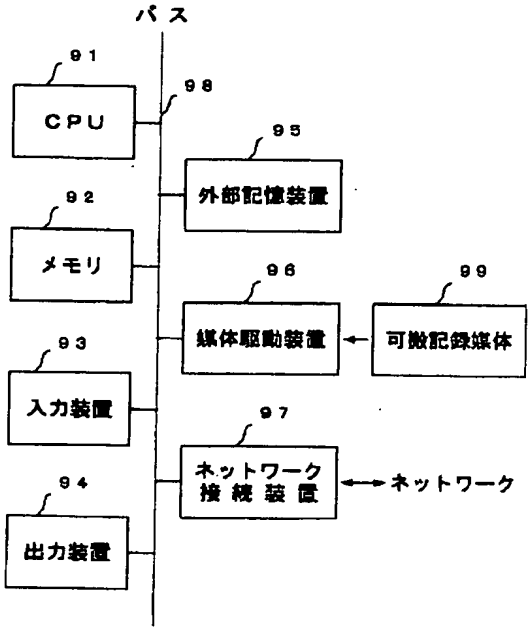
【図１６】

割当処理のフローチャート



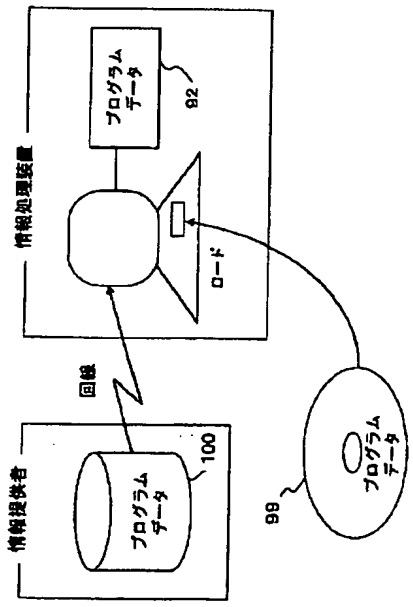
【図１７】

情報処理装置の構成図



【図１８】

記録媒体を示す図



【図２０】

第2のシミュレーション結果を示す図

		待ち行列スケジューリング	階層的スケジューリング	
			δ0=25	δ0=50	δ0=75
総実行時間 [sec]	6.12×10^6	5.62×10^6	4.89×10^6	4.10×10^6	5.01×10^6
性能比（逐次=1）	1	0.91	0.80	0.67	0.82
メモリ使用率 [%]	25	36	55	57	52

【図１９】

第1のシミュレーション結果を示す図

	逐次実行	待ち行列スケジューリング	階層的スケジューリング		
			δ0=25	δ0=50	δ0=75
総実行時間 [sec]	8.56×10^{6}	6.22×10^{6}	5.48×10^{6}	5.62×10^{6}	5.71×10^{6}
性能比(逐次=1)	1	0.72	0.64	0.65	0.66
メモリ使用率 [%]	23	34	52	49	48

【図２１】

第3のシミュレーション結果を示す図

	逐次実行	待ち行列スケジューリング	階層的スケジューリング		
			δ0=25	δ0=50	δ0=75
総実行時間 [sec]	5.68×10^{6}	5.49×10^{6}	5.12×10^{6}	5.03×10^{6}	4.78×10^{6}
性能比(逐次=1)	1	0.97	0.90	0.88	0.84
メモリ使用率 [%]	21	33	48	45	50